CLAIMS

What is claimed is:

1. A task management method for determining optimal placement of task components, said method comprising:

a) generating a communication graph representative of a task, task components represented as nodes of said communication graph and edges connecting ones of said nodes, said edges representing communication between connected nodes and being weighted proportional to communication between connected nodes;

b) assigning terminal nodes to said communication graph;

c) identifying high communication edges within said communication graph, said high communication edges having a weight indicating a communication level exceeding the communication level for a selected terminal node;

d) determining a min cut solution for said communication graph, high communication edges being excluded from determined min cut solutions; and

e) placing task components on said terminal nodes responsive to said min cut solution.

2. A task management method as in claim 1, after the step (b) of assigning terminal nodes, further comprising the step of:

b1) identifying independent nets in said communication graph, each of said independent nets being connected between a plurality of said terminal nodes.

3. A task management method as in claim 2, wherein the step (c) of identifying high communication edges comprises the steps of:

i) summing the weight of terminal edges connected to terminal nodes of an independent net;

ii) identifying the terminal node having the second largest sum as the second heaviest terminal node;

iii) identifying any edge connected to at least one non-terminal node and not connected to said second heaviest node and at least as heavy as the second largest sum; and

iv) collapsing each identified edge.

4. A task management method as in claim 3, further comprising the step of:

v) repeating steps (i) - (iv) until no edges are identified as being heavier than the second largest sum.

5. A task management method as in claim 4, wherein identified edges are selectively collapsed comprising the steps of:

i) merging nodes at opposite end of each identified edge to form a single merged node including the components of both original nodes;

ii) discarding the identified edge; and

iii) replacing groups of parallel edges with a single edge having a weight equal to the sum of parallel edge weights.

6. A task management method as in claim 5, wherein the step (c) of determining a min cut solution comprises the steps of:

i) identifying independent nets in reduced nets;

ii) identifying and collapsing edges selectively identified as being heavier than the second heaviest terminal node in said identified independent nets, said independent nets being further reduced; and

iii) repeating steps (i) - (ii) until a min cut solution has been found.

7. A task management method as in claim 6, wherein each said task component is a unit of the computer program.

Sub G4

8. A task management method as in claim 7, wherein said each computer program unit is an instance of an object in an object oriented program

9. A task management method as in claim 7, wherein in step (d) computer program units are placed on computers, computer program units being placed on a common computer being combined into a single component.

10. A task management method as in claim 6 wherein said task is integrated circuit chip functional element placement and said task components are logic elements, said logic elements being placed on an integrated circuit chip in placement step (e).

11. A distributed processing system for determining optimal placement of computer program components on multiple computers, said distributed processing system comprising:

means for generating a communication graph of nodes interconnected by edges and representative of a computer program, computers executing said computer program being represented as terminal nodes, computer program components being represented as non-terminal nodes, said edges representing communication between connected nodes and being weighted proportional to communication between connected nodes;

means for summing the weight of edges connected to terminal nodes;

means for identifying a second heaviest terminal node;

means for comparing edges with the sum for said second heaviest terminal node;

means for determining a min cut solution for said communication graph, edges heavier than said sum being excluded from determined min cut solutions responsive to said comparison; and

means for placing program components on ones of said computers responsive to said determined min cut solution; and

said computer program being executed by said computers.

12. A distributed processing system as in claim 11, further comprising:

means for identifying independent nets connected between a plurality of said terminal nodes.

13. A distributed processing system as in claim 12, further comprising:

means for collapsing said edges heavier than said sum.

14. A distributed processing system as in claim 13, wherein the means for identifying edges heavier than said sum comprises:

means for summing the weight of terminal edges connected to terminal nodes;

means for identifying the terminal node having the second largest sum as the second heaviest terminal node;

means for comparing edge weights against said second largest sum; and

means for selectively collapsing edges identified as having a weight at least as heavy as the second largest sum.

15. A distributed processing system as in claim 14, the means for selectively collapsing edges further comprising:

means for merging nodes on either end of a selected edge and discarding said selected edge; and

means for replacing pairs of parallel edges attached to said merged node with a single edge.

Sub 94

16. A distributed processing system as in claim 15, wherein the means for comparing edge weights further comprises:

means for selecting edges attached to at least one non-terminal node and not attached to said second heaviest terminal node.

17. A distributed processing system as in claim 16, wherein each said program component is a unit of the computer program.

18. A distributed processing system as in claim 17, wherein said each program unit is an instance of an object in an object oriented program

19. A computer program product for determining optimal placement of functional components, said computer program product comprising a computer usable medium having computer readable program code thereon, said computer readable program code comprising:

computer readable program code means for generating a communication graph of nodes interconnected by edges and representative of a function, a plurality of said nodes being terminal nodes, functional components being represented as non-terminal nodes, said edges representing communication between connected nodes and being weighted proportional to communication between connected nodes;

computer readable program code means for summing the weight of edges connected to terminal nodes;

computer readable program code means for identifying a second heaviest terminal node;

computer readable program code means for comparing edges with the sum for said second heaviest terminal node;

Sub 94

computer readable program code means for determining a min cut solution for said communication graph, edges heavier than said second heaviest edge being excluded from determined min cut solutions responsive to said comparison; and

computer readable program code means for placing functional components responsive to said determined min cut solution.

20. A computer program product as in claim 19, further comprising:

computer readable program code means for identifying independent nets connected between a plurality of said terminal nodes.

21. A computer program product as in claim 20, further comprising:

computer readable program code means for collapsing edges heavier than said sum.

22. A computer program product as in claim 21, wherein the computer readable program code means for identifying edges heavier than said sum comprises:

computer readable program code means for summing the weight of terminal edges connected to terminal nodes;

computer readable program code means for identifying the terminal node having the second largest sum as the second heaviest terminal node;

computer readable program code means for comparing edge weights against said second largest sum; and

computer readable program code means for selectively collapsing edges identified as having a weight at least as heavy as the second largest sum.

23. A computer program product as in claim 22, wherein the computer readable program code means for selectively collapsing edges further comprising:

computer readable program code means for merging nodes on either end of a selected edge and discarding said selected edge; and

computer readable program code means for replacing pairs of parallel edges attached to said merged node with a single edge.

24. A computer program product as in claim 23, wherein the computer readable program code means for comparing edge weights further comprises:

computer readable program code means for selecting edges attached to at least one non-terminal node and not attached to said second heaviest terminal node.

25. A computer program product as in claim 24, wherein said function is a computer program and each said functional component is a unit of the computer program.

26. A computer program product as in claim 25, wherein each said program unit is an instance of an object in an object oriented program.

27. A computer program product as in claim 24 wherein said function is an integrated circuit chip and said functional components are logic elements.